Comunicação entre micros via TCP/IP

O TCP/IP é o protocolo de comunicação usado na Internet. Ele pode ser usado para permitir que dois programas troquem informações ente si. E o melhor: os programas podem estar tanto na mesma máquina como conectados via Internet.

Em várias situações precisamos trocar informações entre programas. Existem muitas maneiras de fazer isto, provavelmente a mais comum é via tabelas de bancos de dados. No entanto, uma das técnicas se destaca: troca de informações via TCP/IP. Esta técnica tem as seguintes vantagens:

- É relativamente fácil de usar;
- Funciona para programas rodando no mesmo micro, em rede local ou até mesmo via Internet;
- É rápida e eficiente. A única desvantagem é que o protocolo TCP/IP deve estar instalado na máquina. Como este protocolo é necessário para o acesso à Internet e a Internet é muito usada, este problema é muito pequeno.

O protocolo TCP/IP

Para transmitir informações de um lado para outro da rede, a Internet se baseia em diversos protocolos. Os principais protocolos, sob os quais todos os demais são construídos são: TCP ou Transmission Control Protocol; IP ou Internet Protocol. Estes dois protocolos são usados em conjunto e conhecidos coletivamente como TCP/IP. O IP tem a responsabilidade de enviar dados de um computador a outro. Cada transmissão IP traz inclui os campos mostrados na figura1.

Cabeçalho (checksum, contador de expiração etc)
Endereço IP de origem Endereço IP de destino Dados
Figura 1

O campo de dados contém a informação que deve ser enviada. Caso o campo de dados seja muito grande para uma rede em particular, o IP pode quebra-lo em vários pacotes e remonta-los no destino. O endereço IP é um número de 32 bits, necessário para qualquer troca de informações pela Internet. Cada equipamento conectado à Internet deve ter um endereço IP único. Este endereço pode ser atribuído basicamente de duas formas: Fixamente: Neste caso, cada equipamento tem sempre o mesmo número IP, atribuído durante a configuração. É normalmente o caso de universidades e grandes empresas americanas que se conectaram cedo à Internet e pegaram uma grande quantidade de endereços IP. Dinamicamente: Neste caso, o computador recebe um endereço IP "emprestado" enquanto estiver conectado. Assim que ele se desconectar, um outro computador pode usar o endereço. É o método normalmente usado em acesso via linha discada a um provedor. É costume especificar o endereço IP como quatro números decimais de 0 a 255, por exemplo: 207.68.156.73 (www.microsoft.com). O algarismo final tem dois valores reservados: no início e no fim da faixa, normalmente 0 e 255, que jamais correspondem a nenhum equipamento. Os endereços IP são difíceis para os seres humanos decorarem. Por esta razão, foi criada uma espécie de "lista telefônica" que atribui sinônimos em forma de caracteres a endereços IP. Por exemplo, www.ramosdainformatica.com.br é um "sinônimo" de IP. Esta "lista telefônica" usa um protocolo chamado "DNS". A base de dados do DNS é mantida pelos órgãos gestores da Internet (no Brasil é a FAPESP). A "publicação na lista" custa uma taxa anual que hoje está em R$70,00. O TCP usa o protocolo IP, mas garante que os dados chegaram ao seu destino e também que chegam na ordem correta. Se houver um erro na transmissão, o pacote é enviado novamente. O TCP é capaz de manter uma conexão entre as duas pontas da comunicação. Toda a comunicação via TCP/IP é feita através de "portas". Uma porta é um numero de 16 bits que identifica um destino dentro do servidor. Ou seja, para estabelecer comunicação entre dois computadores precisamos a porta, além do endereço IP. Algumas portas tem protocolos normalmente associados a elas, por exemplo: FTP: 21 Telnet: 23 SMTP (Envio de correio): 25 POP3 (Recepção de correio): 125 HTTP (Web) 80 NNTP (News): 119 Quando formos criar um programa para troca de informações via TCP/IP, devemos escolher uma porta que não não esteja sendo usada. É uma boa idéia permitir que o usuário troque a porta de alguma forma, para o caso de conflitos. Outra coisa: para usar na Internet, devemos configurar os roteadores e "firewalls" no caminho para aceitarem conexões na porta escolhida.

TCP/IP no Delphi

As versões 3 e 4 do Delphi vem com dois componentes destinados a implementar conexões TCP/IP: *TClientSocket* e *TServerSocket*. Estes componentes podem ser usados de duas maneiras: "blocking" e "non-blocking". A primeira maneira dispara uma nova "thread" para cada comunicação, a segunda roda toda a comunicação na mesma thread. É muito mais fácil programar como "non-blocking" e os exemplos mostrados a seguir usam este modelo. Caso você decida por "blocking", leia atentamente a documentação do Delphi sobre o assunto. *TServerSocket* implementa o lado "servidor" de uma conexão TCP/IP. Para usa-lo, devemos ajustar a porta de comunicação e chamar o método *Open*. O componente então fica "escutando" a porta por uma conexão. Depois de efetuada uma conexão, o evento *OnClientRead* é disparado quando existirem dados a serem lidos.

Veja o exemplo ESCUTA.DPR:
Veja o código do programa acima:

```
procedure TForm1.BtEscutaClick(Sender: TObject);
begin
  // Abre porta para escutar
  ServerSocket1.Port := StrToInt(EdPorta.Text);
  ServerSocket1.Open;
  BtEscuta.Enabled := false;
  BtDesconecta.Enabled := true;
end;

procedure      TForm1.ServerSocket1ClientRead(Sender:      TObject;      Socket:
TCustomWinSocket); .
var
  S1, S2: string;
begin
  // Lê bytes da conexão, mexe um pouco e devolve
  S1 := Socket.ReceiveText;
  S2 := 'Você enviou >' + S1 + '<'#13#10;
  Socket.SendText(S2);
  Memo1.Lines.Add('Recebido:' + S1 + ' Devolvido:'+ S2);
end;

procedure TForm1.BtDesconectaClick(Sender: TObject);
begin
  // Fecha conexão
  ServerSocket1.Close;
  BtEscuta.Enabled := true;
  BtDesconecta.Enabled := false;
end;

procedure TForm1.ServerSocket1ClientConnect(Sender: TObject; Socket:
TCustomWinSocket);
begin
  Memo1.Lines.Add('Conectado com:' + Socket.RemoteAddress);
end;

procedure TForm1.ServerSocket1ClientDisconnect(Sender: TObject; Socket:
TCustomWinSocket);
begin
  Memo1.Lines.Add('Desconectado de :' + Socket.RemoteAddress);
 end;
```

Podemos testar o programa acima com o programa TELNET, que vem com o Windows.
Rode programa ESCUTA e clique no botão "Escuta". A seguir, rode o Telnet na mesma máquina, peça para conectar e selecione as seguintes opções: "localhost" é um sinônimo TCP/IP para a própria máquina.

Poderíamos também ter usado o endereço IP 127.0.0.1 que tem o mesmo significado. Se você tiver dois micros conectados em rede, poderá rodar cada programa em uma máquina, ajustando no cliente o nome ou endereço IP da máquina na qual o programa servidor está rodando. Uma vez estabelecida a conexão, o que você digitar no Telnet irá até o servidor, que enviará uma resposta. Podemos também escrever um programa para fazer o papel de cliente, com o componente *TClientSocket*. Para usar este componente, você deve ajustar as propriedades *Port* com o número da porta TCP/IP e também *Address* ou *Host* para identificar o computador remoto. *Address* aceita uma string no format X.X.X.X como um endereço IP. Já *Host* aceita um nome como string.

O exemplo FALA.DPR se conecta com o exemplo anterior, ESCUTA.

Acesso a banco de dados

O exemplo a seguir (EMPINFO.DPR) faz algo mais útil: consulta uma tabela de banco de dados e retorna uma resposta. Ele usa a tabela EMPLOYEE.DB do alias DBDEMOS. Ele recebe "EmpNo", a chave primária da tabela e retorna o nome e sobrenome da pessoa. Neste exemplo, temos algumas complicações: Devemos esperar receber um "enter" para efetuar a pesquisa. O programa pode estar servindo várias conexões simultaneamente. Por causa disto, temos que armazenar uma string com o comando para cada conexão existente. Felizmente, existe a propriedade *Data* na conexão TCP/IP exatamente para isto.

Veja o código do programa:

```
type PString = ^string;

procedure TForm1.ExecutaCmd(S: string; Socket: TCustomWinSocket);
var
  Resposta: string;
Begin
 try
   // Procura o registro
  if Table1.Locate('EmpNo', S, [])
  then Resposta := Table1FirstName.Value + ' ' + Table1LastName.Value
  else Resposta := 'Não Achado';
except
  // Retorna mensagem de erro, caso tenha ocorrido
  on E:Exception do Resposta := E.Message;
end;
  Socket.SendText(Resposta + #13#10);
end;

procedure TForm1.BtEscutaClick(Sender: TObject);
begin
  // Abre porta para escutar
  ServerSocket1.Port := StrToInt(EdPorta.Text);
  ServerSocket1.Open;
  BtEscuta.Enabled := false;
  BtDesconecta.Enabled := true;
end;

procedure TForm1.ServerSocket1ClientRead(Sender: TObject; Socket:
TCustomWinSocket);
var
  S: PString;
  P: integer;
  Cmd: string;
begin
  // Pega e concatena dados novo
  S := PString(Socket.Data);
  S^ := S^ + Socket.ReceiveText;
```

```
  // Procuta fim de linha
  P := pos(#13, S^);
  if P <> 0
  then begin
         // Separa comando e executa
         Cmd := copy(S^, 1, P - 1);
         S^ := copy(S^, P + 1, length(S^) - P);
         ExecutaCmd(Cmd, Socket);
       end;
end;

procedure TForm1.BtDesconectaClick(Sender: TObject);
begin
  // Fecha conexão
  ServerSocket1.Close;
  BtEscuta.Enabled := true;
  BtDesconecta.Enabled := false;
end;

procedure TForm1.ServerSocket1ClientConnect(Sender: TObject; Socket:
TCustomWinSocket);
var
  S: PString;
begin
  AcrescentaMemo(Memo1, 'Conectado com:' + Socket.RemoteAddress);
  // Aloca string que guardará o comando
  S := new(PString);
  S^ := '';
  Socket.Data := new(PString);
end;

procedure TForm1.ServerSocket1ClientDisconnect(Sender: TObject; Socket:
TCustomWinSocket);
begin
  // Libera memória dispose(Socket.Data);
  AcrescentaMemo(Memo1, 'Desconectado de :' + Socket.RemoteAddress);
end;
```

Podemos testar o programa acima tanto com o TELNET como com o projeto CONSULTAEMP, feito especialmente para isto. Como pudemos ver, é relativamente fácil escrever programas que se conectam, diretamente via TCP/IP. Copie aqui o Programa Exemplo

Referência: Este programa, assim como outros exemplos, podem ser encontrados nos exemplos do Delphi 4, que acompanha o próprio Delphi.